Section 3
Processing Overview

3.1 Summary

The ETMS processing functions are organized in three groups, as shown in Figure 3-1.

The *ETMS Central Functions* are real-time, continuously running functions that are located at the Volpe Center and support the entire ETMS network. The *ETMS Central Functions* process the real-time incoming data, maintain a large, distributed database, perform the traffic modeling, and transmit processed data to the field sites where traffic management is performed.

The *ETMS Field-site Functions* are located at every field site where traffic management is performed. The *ETMS Field-site Functions* run continuously. The *ETMS Field-site Functions* maintain local databases of data received from the *ETMS Central Functions* and make the ETMS data available to the traffic management specialist through the *TSD*.

The *Auxiliary Support Functions* include off-line functions that are run periodically when data is received. The *Auxiliary Support Functions* produce static databases and files for data such as the airline schedule data and geographical data. The *ETMS Central Functions* and *ETMS Field-site Functions* use the static data during the real-time processing. These *Auxiliary Support Functions* are run only at the Volpe Center. Other *Auxiliary Support Functions* are the *Ground Time Prediction Subsystem* and *Run-Time Support*.

The general approach to the ETMS architecture is to perform the processing at the central site and distribute the resultant data to the field sites. Centralizing the processing saves greatly on the computing resources required. Continuously distributing the data to the field sites provides the best response time to traffic management specialist requests for data. However, this approach is not followed rigidly. Some databases have been centralized because of their very large sizes and the relatively low frequency of data access.

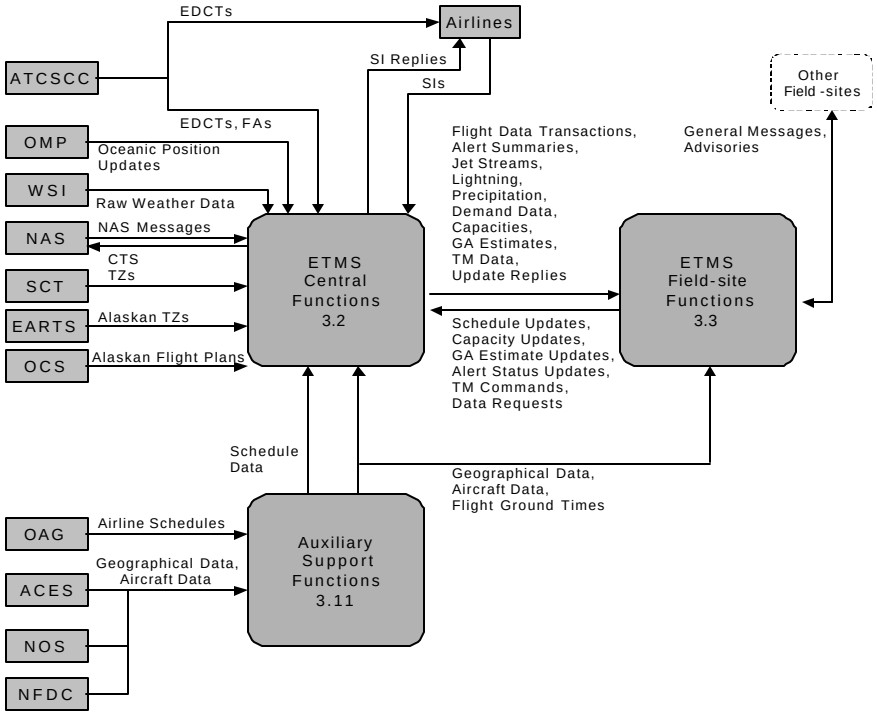

Figure 3-1. Data Flow of the ETMS

The ETMS continuously distributes all data that is pertinent to individual flights for the next 12 hours to each field site as flight data transactions. The ETMS also continuously distributes alert summaries, jet streams, precipitation, and lightning data.

The ETMS maintains all other data only at the central site. This data includes schedule data beyond the next twelve hours, traffic demands for airports, sectors, and fixes, capacities, general aviation (GA) estimates, traffic management data concerning ground delay programs, and weather reports (METAR and TAF). The *ETMS Field-site Functions* answer a traffic management specialist's request for this data by requesting the data from the central site. The specialist can also update data on the central site databases. When a traffic management specialist enters capacities, GA estimates, alert status changes, airline schedule updates at a field site, or a traffic management command, the *ETMS Field-site Functions* send the values to the *ETMS Central Functions*, which update the databases and send back a reply.

3.2 ETMS Central Functions

The *ETMS Central Functions* are organized in several groups as shown in Figure 3-2.

The *ETMS Communications Functions* handle all communications between the ETMS central and field sites. The *ETMS Communications Functions* distribute certain types of data automatically to *User Support Functions* at all field sites; these data are the flight data transactions, alert summary data, jet streams, precipitation, and lightning. The *ETMS Communications Functions* pass commands and data requests entered by traffic management specialists through *User Functions* and *User Support Functions* at field sites to the appropriate functions at the central site; the *ETMS Communications Functions* also send update replies (SAs, FTs, TM data, TM replies, detailed alert data, flight lists, etc.) from the central functions back to the requesting sites. The commands and data requests that are transmitted from the field sites are demand data requests, schedule updates, schedule data requests, capacity updates, capacity requests, GA estimate updates, GA estimate requests, TM requests, TM commands, and alert status updates.

The *ETMS Communications Functions* also handle all message traffic among ETMS components. The components can be located in geographically diverse positions. The *ETMS Communications Functions* implement a Wide Area Network (WAN) form of message switching that utilizes Local Area Network (LAN) technologies wherever possible. Some external interfaces that are not suitable to this unified WAN technology are handled by the *External Communications Functions* described below.

> **NOTE:** Message traffic between components at the same site are shown as not going through the ETMS Communications Functions in Figure 3-2 to maintain the simplicity of the diagram.

The *External Communications Functions* handle communications with systems that are external to the ETMS. The external communications components do not use the standard ETMS communications protocols and therefore require specialized interface converters. The *External Communications Functions* receive NAS messages from the NAS, oceanic position updates (TOs) from the *Offshore Message Processor* (*OMP*), Estimated Departure Clearance Times (EDCTs) and Fuel Advisory messages (FAs) from the ATCSCC, weather data from Weather Systems Incorporated (WSI), and substitution requests (SIs) from the airlines. The *External Communications Functions* send control times (CTs) to the NAS and SI replies to the airlines. The *External Communications Functions* send the NAS messages, TOs, and grid winds weather data to the *Traffic Model Functions*, send the EDCTs, FAs, and SIs to the *TM Functions*, and send other weather data to the *ETMS Communications Functions*. The *External Communications Functions* also receive schedule messages from the *SDB Functions* and send them to the *Traffic Model Functions*. The schedule messages are routed this way to include them in the stream of NAS messages sent to the *Traffic Model Functions*.

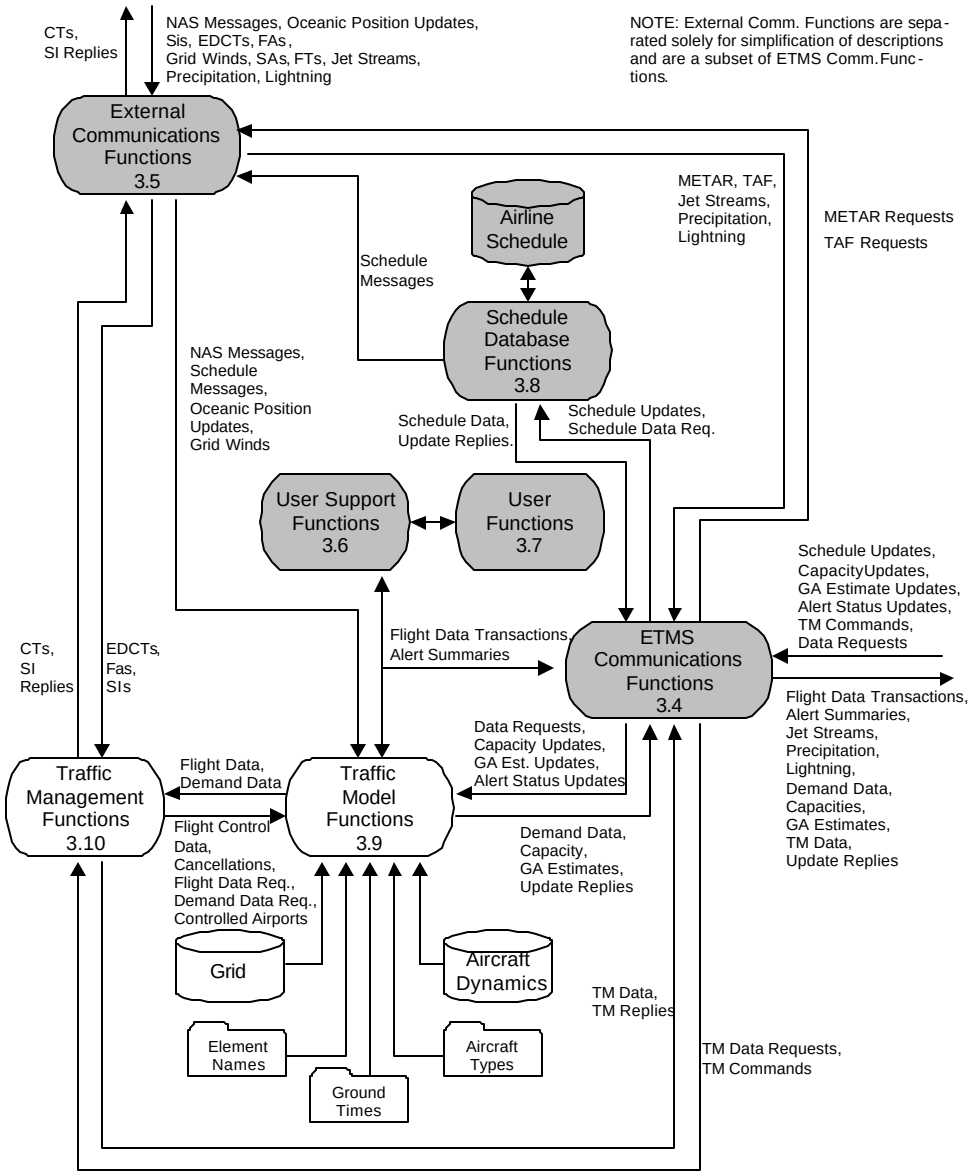

Figure 3-2. Data Flow of the ETMS Central Functions

The main role of the *User Support Functions* is to maintain and display data for the traffic management specialist. This is performed at the field site. The *User Support Functions* also exist at the central site to support system monitoring and testing.

The *SDB Functions* integrate the airline schedule data into the ETMS. The *Auxiliary Support Functions* (not shown) periodically build an airline schedule database from data received from the OAG. The *SDB Functions* use the airline schedule database to feed *schedule messages* for flights that are scheduled to depart within 12 hours of the current time on the current day to the *External Communications Functions*.

The *SDB Functions* also perform maintenance of the airline schedule database. Traffic management specialists may enter schedule updates to add/edit, cancel, inhibit, or activate scheduled flights through the field-site *User Functions*. The field-site *User Support Functions* send the updates through the *ETMS Communications Functions* to the *SDB Functions*, which modify the database accordingly and send a reply to the specialist. If the database change affects a flight that has already been sent to the *External Communications Functions*, the *SDB Functions* send a schedule message to indicate the change.

The *SDB Functions* also process schedule data requests from field-site *User Functions* to support the request reports. The *SDB Functions* look up the schedule data for the requested time interval, date, and airport; and send the data back to the field-site *User Functions* that made the request.

The *Traffic Model Functions* perform the detailed traffic modeling for ETMS. The *Traffic Model Functions* use the following dynamic data received from the *External Communications Functions*:

- NAS messages — messages generated by the NAS computers including flight plans, departures, arrivals, etc.
- Schedule messages — messages generated by the *SDB Functions* to feed scheduled airline flights for the next 12 hours into the traffic modeling
- Oceanic position updates — messages obtained from OMP providing position updates for oceanic flights
- Grid Winds — data describing the winds aloft

The *Traffic Model Functions* use the following dynamic data received from the *TM Functions:*

- Flight control data — controlled departure times from EDCTs and SIs
- Cancellations — flight cancellations from SIs

The *Traffic Model Functions* use five static data sources produced by the *Auxiliary Support Functions*:

- Grid database — geographical data for flight path processing
- Aircraft dynamics database — data for modeling flight ascent and descent profiles
- Element names file — names of airports, fixes, and sectors
- Aircraft types file — definitions of all aircraft designators
- Departure Ground times file — estimated times from gate pushback to wheels up based on historical data

The *Traffic Model Functions* interpret all incoming data and saves it in internal databases. The *Traffic Model Functions* use the input data to maintain a best estimate of the performance of each flight and the traffic demands at each element. The *Traffic Model Functions* send flight data transactions and alert summaries to the local *User Support Functions* and through the *ETMS Communications Functions* to the remote *User Support Functions*. The flight data transactions contain data from the incoming messages along with processed flight data that is not directly available from the incoming messages, including the waypoints of the flight path and estimated departure and arrival times. The alert summaries contain summary demand and capacity values for all alerted airports, sectors, and fixes. The *Traffic Model Functions* provide flight and demand data to the *TM Functions.*

The *Traffic Model Functions* process commands and data requests entered by traffic management specialists through field-site *User Functions*. The commands allow a traffic management specialist to update capacities for any airport, sector, or fix; to update GA estimates for any airport; to change alert statuses; to update the schedule database; or to purge a ground delay program. The data requests allow a traffic management specialist to see existing capacities or GA estimates. (A traffic management specialist can cause the *User Support Functions* to make demand data requests by requesting monitor/alert data and request reports.)

The *TM Functions* implement controlled departure times determined by a traffic management specialist at the ATCSCC by issuing control times to the NAS. The *TM Functions* also apply changes to the control times issued by the airlines according to traffic management policy. The *TM Functions* receive EDCTs, FAs, and SIs from the *External Communication Functions* and receive flight data and demand data from the *Traffic Model Functions*. The *TM Functions* send flight control data, flight and demand data requests, and cancellations to the *Traffic Model Functions* and send SI replies and CTs to the *External Communications Functions*. The *TM Functions* allow a specialist to request control data and cancel flight controls. The *ETMS Communications Functions* send TM commands and data requests to the *TM Functions,* which in turn send back TM data and replies.

3.3 ETMS Field-site Functions

The *ETMS Field-site Functions* are organized into two groups as shown in Figure 3-3.

The *ETMS Communications Functions* perform all communications with the central site. All data available to the traffic management specialist through the *User Support Functions* is transmitted from the central site through the *ETMS Communications Functions*. The *ETMS Communications Functions* send all data received from the central site to the *User Support Functions*, and all data requests, updates, and TM commands received from the *User Support Functions* are sent to the central site.

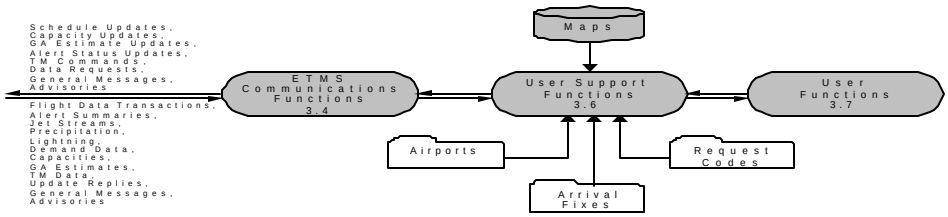

Figure 3-3. Data Flow of the ETMS Field-site Functions

The *User Support Functions* receive flight data transactions, alert summaries, jet streams, precipitation, and lightning automatically; i.e., the *User Support Functions* do not request these types of data. The *User Support Functions* maintain these types of data in tables at the field site and make them available to the traffic management specialist through the *TSD*. The *User Support Functions* request the remaining data from the central site as needed to support a traffic management specialist's data requests. The *User Support Functions* also use data from the following four static data sources produced by the *Auxiliary Support Functions*:

- Maps database — geographical data used to draw map overlays on the screen
- Airports file — the name and location of each airport, used to *ghost* flights towards their destinations

- Codes file — many character codes (e.g., airport names, airline names, and aircraft designators) used to process requests for list/count reports
- Arrivals fixes — data used to associate arrival fixes with airports

The *User Functions* provide a variety of graphical and textual displays of the data, as described in Section 2, Functional Overview. The *User Support Functions* save replay files, which allow a traffic management specialist to replay aircraft situation data from the past. The *User Support Functions* can support multiple traffic management workstations at each site.

3.4 ETMS Communications Functions

The *ETMS Communications Functions* are responsible for exchanging messages between any two processes anywhere within the ETMS Wide Area Network (WAN). A typical central site setup is shown in Figure 3-4, and a representative field site is shown in Figure 3-5. The communications functions are highlighted in white.

The ETMS communications software allow any process at any ETMS site to communicate with any other process at any ETMS site. The communications are provided through a network addressing scheme which represents the ETMS WAN at three levels: *sites*, *nodes*, and *processes*. A *site* is a Local Area Network (LAN) of ETMS workstations; a TMU is one site (e.g., ZBW), the ATCSCC is two sites (CFCA and CFCB), and the ETMS central site is two sites (ETMSA and ETMSB). A *node* is an ETMS workstation on the LAN (e.g., //wkstn01). A *process* is a computer program running on a node (e.g., TSD). The communications for a node are handled by a single *Node Switch* process. All processes on that node must be connected to the *Node Switch* to use the ETMS communications (Figures 3-4 and 3-5 show the connection of ETMS processes to the *Node Switches*). The communications for a site are handled by a single *Site Switch* process. All nodes at that site must be connected to the *Site Switch* to use the ETMS communications.

A process sends a message to another process by attaching a network address to the message and sending it to the local *Node Switch*. The *Node Switch* determines whether the destination address is another process on the same node. If so, it passes the message directly to the destination process; if not, it passes the message to the local *Site Switch*. Similarly, the *Site Switch* determines whether the destination address is a node at the same site. If so, it passes the message to the *Node Switch* on the destination node; if not, it sends the message to a *Site Switch* at the destination site.

A process does not need to know the exact address in order to send messages or get data. The ETMS communications software provides a set of pre-defined services, which a process can address. When a process sends a message to a service, the communications software will find the nearest instance of the service and send the message there. The service might exist on the same node, another node at that site, or at another site.

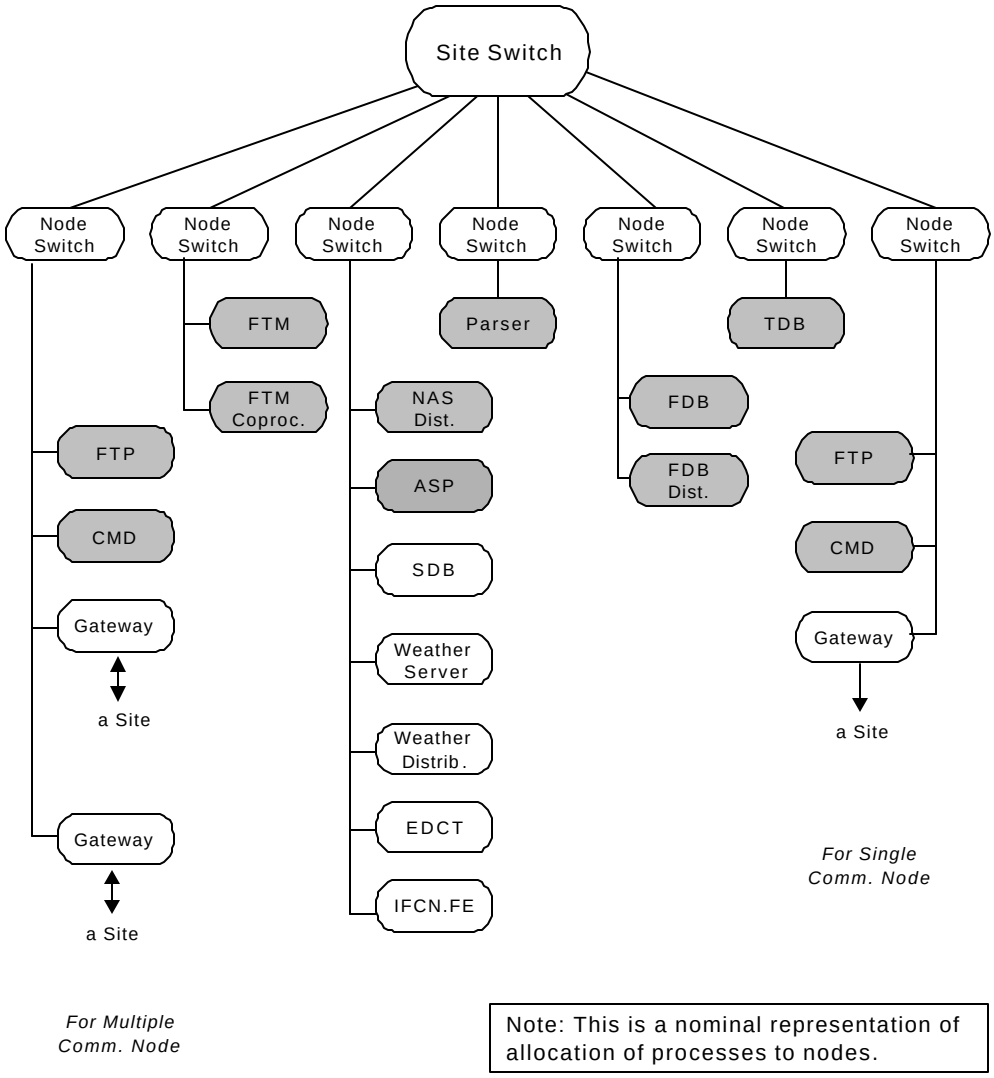

Figure 3-4. Data Flow of the ETMS Communications Functions – Central Site

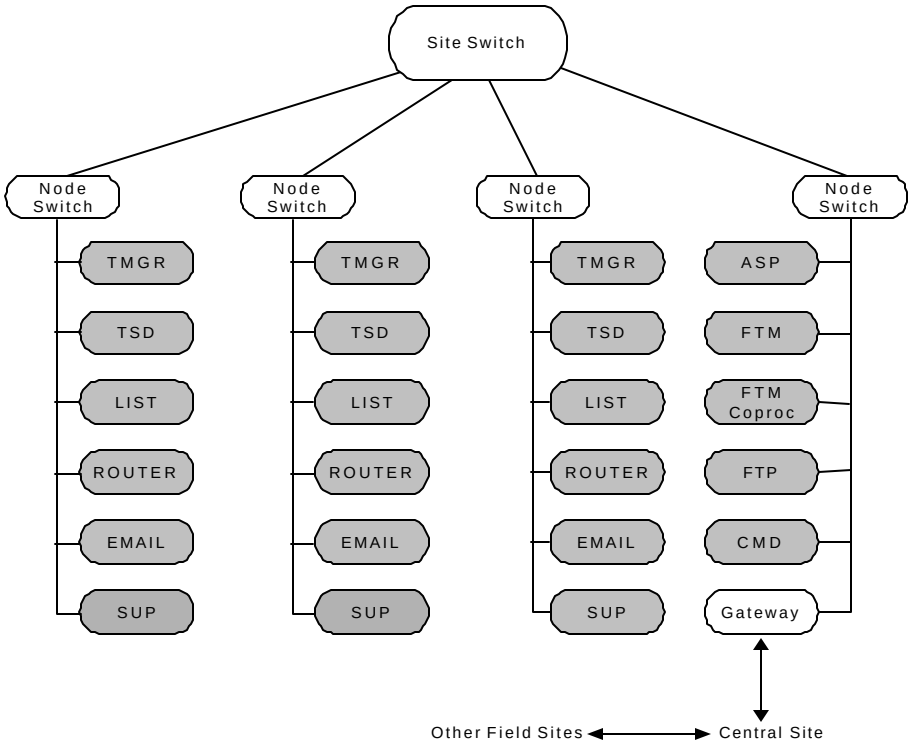

Figure 3-5. Data Flow of the ETMS Communications Functions – Field Site

The *Router* is a function that connects to a *Node Switch*. The *Route*r allows the specialist to send and receive messages at the *ETMS User Functions*. It is responsible for the notification of all messages sent from the *EMail* function destined for another *Router*, received messages sent from another *Router*, and any error messages encountered. The *Router* is created as a server process, which remains running after a user is logged out. The *Router* function is represented by an icon on the user's workstation. Initialization and information messages are defaulted to a low priority. All received messages are set to a medium priority, and error messages are set to a high priority. The *Router* must be connected to a Node Switch to allow

the *Router* to communicate with *EMail* for message transmission through the ETMS communications software. Since the *Router* is used by *EMail* to transmit messages through *ETMS*, the *Router* must run on every workstation used to send or receive messages.

The communications path between two sites depends on which sites they are and is determined by the configuration of the physical communications links. Each ETMS field site is connected to the ETMS central site by landline. When the ETMS central site and an ETMS field site communicate, the *Site Switches* can pass messages using a direct connection. However, when a *Site Switch* at an ETMS field site sends a message to another ETMS field site, there is no direct connection. The field *Site Switch* in this case sends the message to the central *Site Switch*, which then relays the message to the destination field *Site Switch*.

The ETMS central site communicates with each field site using a dedicated 56 kbps fiber-optic link. A communication *Gateway* process handles the communications for each site. There are two configurations for communications nodes at the ETMS central site (see Figure 3-4). The original design, the *Single Comm. Node*, calls for a one-to-one correspondence; that is, one communications node is reserved to communicate with one field site. The more recent design, the *Multiple Comm. Node*, allows one communications node with special hardware to communicate with four field sites. At the field site (see Figure 3-5), the communications gateway runs on the file server node.

There is a single data link line to the remote site. It connects to an A/B switch, which is used to select which of two routers will be routing the message traffic. The second router will be running but not routing. The field site routers connect to a 10Mb/100Mb switch (for network compatibility). Each workstation is connected to this 10Mb/100Mb switch.

The types of messages carried on the communications lines from the central site to a field site are flight data transactions, alert summaries, and weather data. A field site typically sends request/reply messages for report data. Users at any site are allowed to send messages and files to users at any other site. Files are restricted to certain directories and file type (ASCII). Security functions allow other files to be transferred under proper authorization.

The communications functions support logistical functions such as file transfer and remote command execution. The remote command execution is used by ETMS operations personnel to diagnose and correct system problems.

A dedicated process, the *Flight Database (FDB) Distribution* process, handles the distribution of database updates from the ETMS central site to the ETMS field sites (see Figure 3-6). The *FDB Distribution* process receives flight data transactions from the central-site *Traffic Model Functions* and sends them to registered *User Support Functions* at the field sites.

Figure 3-6. Data Flow of the FDB Distribution Process

13.5 External Communications Functions

External Communications Functions handle external components that do not comply with the ETMS WAN packet switching technologies and therefore require a protocol driver and converter to interface with the ETMS. The *External Communications Functions* are shown in Figure 3-7. All communications between the *External Communications Functions* and other ETMS functions are done through the *ETMS Communications Functions*.

The ARTCCs send NAS data directly to the ETMS central site. The *NAS Distribution* processes at the central site combine the NAS messages with schedule messages (received from the *SDB Function*) and TOs (from the *OMP Parser*). The combined messages are sent to the *Parser* function on each processing string at the ETMS central site. The use of redundant workstations and communications links (both on land, using redundant fiber-optic links) ensures no service interruption from hardware or communications transmission links.

The *IFCN Interface* (also known as the *IFCN Front End*) receives EDCTs and FAs from *Autosend*. Traffic management specialists generate EDCTs and FAs at the ATCSCC using the Selected Controlled Departure Time (SCDT) function and transmit EDCTs and FAs using the *Autosend* function. The *Autosend* function sends copies of the EDCT and FA files to the ARTCCs, the airlines, and the ETMS central site through the IFCN. The *IFCN Interface* process at the central site notifies the *TM Functions* whenever a file is received. The *TM Functions* send out CTs based on the EDCTs and flight data; the CTs go through the *IFCN Interface* function to the ARTCC host computers.

The airlines generate SIs in response to EDCTs and send them out over ARINC. The *ARINC Interface* function receives the SI messages and sends them to the *TM Functions*. The *TM Functions* generate SI replies, which return through the *ARINC Interface* function to the airlines, with copies to the ATCSCC printer.

The *ARINC Interface* function also receives oceanic position updates from *OMP* over the ARINC line. The *ARINC Interface* sends the oceanic position updates to the *OMP*, which converts the updates into TOs and sends them to the *NAS Distribution* processes.

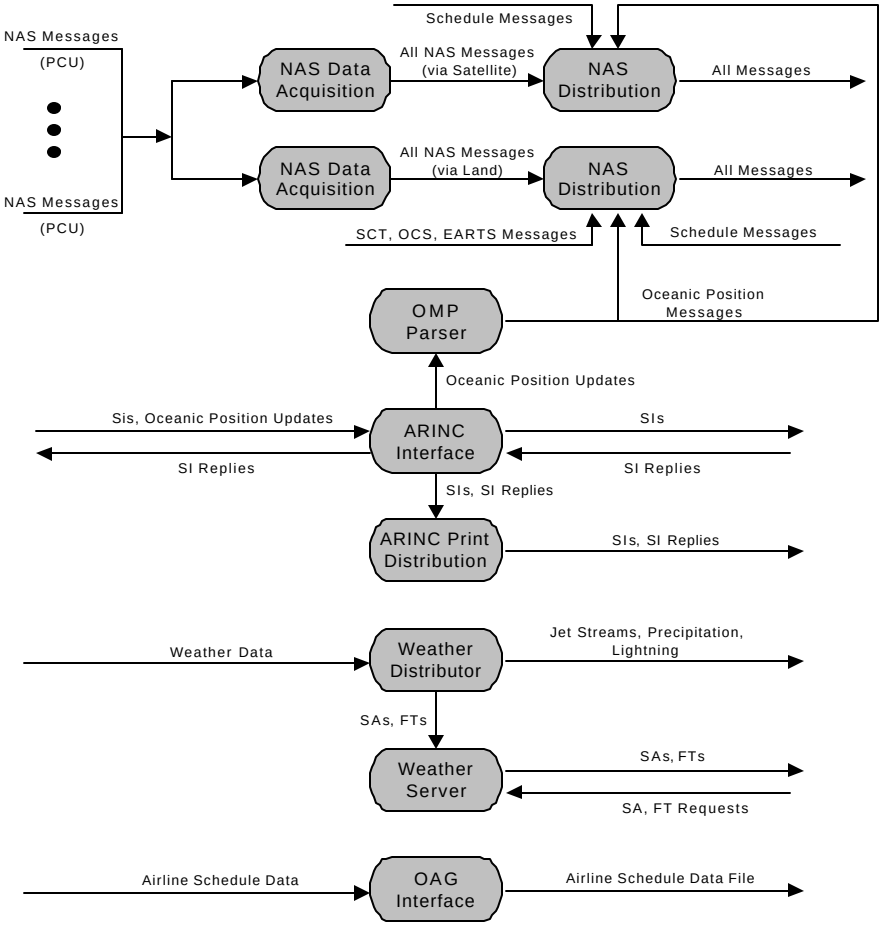

Figure 3-7. Data Flow of the External Communications Functions – Central Site

The weather distribution functions include a *Weather Distributor* and a *Weather Server*. WSI sends raw weather data via satellite to the Volpe Center, where the required weather data is generated. The required weather data includes METAR (files generated hourly), TAF

(generated every nine hours), grid winds (generated every three hours), jet stream graphics (generated every 3 hours), NOWRAD6 precipitation graphics (generated every 5 minutes), and lightning (generated every 5 minutes).

The *Weather Distributor* automatically forwards the jet stream, precipitation, and lightning data to the *ETMS Communications Functions*. The *Weather Distributor* also sends the SAs and FTs to the *Weather Server*. The *Weather Server* stores SA and FT data at the central site and sends METAR and TAF data (reports) in response to requests received from the *ETMS User Functions.*

Airline schedule data is received once a week from the OAG data distribution center in Illinois. The *OAG Interface* process monitors a line used for this purpose. When OAG is ready to transmit data, it connects through a phone line. The *OAG Interface* process receives the airline schedule data in a stream and stores it in an airline schedule data file, which is used by the *Auxiliary Support Functions* to build the airline schedule database.

13.6 User Support Functions

The *User Support Functions* maintain ETMS data at each site and provide the traffic management specialist access to the data through a graphic interface (see *User Functions* in section 3.7). The *User Support Functions* are shown in Figure 3-8.

The *Flight Table Manager* (*FTM*) and the *Alert Server Process* (*ASP*) receive data from the ETMS central site and maintain this data on the *file server* at the site. The *User Functions* and the *Listserver* run on a traffic management specialist's *workstation* and interact with the traffic management specialist to provide requested data displays. The *User Functions* and the *Listserver* are duplicated on every traffic management specialist workstation at a site. A single copy of the *FTM* and *ASP* is required to support all copies of the *User Functions* and *Listserver*.

The *FTM* maintains data about individual flights in a flight table stored on the file server. The *FTM* receives flight data transactions from the central site *Traffic Model Functions*. The central site *Traffic Model Functions* send the flight data to the field sites through the *ETMS Communications Functions*.

The *FTM Coprocessor* handles requests for flight data. The *FTM Coprocessor* receives requests for flight data from the *Listserver* and the *User Functions*, looks up the flights in the flight table, and returns the data to the requesting process.

If, while looking up a flight, the *FTM Coprocessor* determines that a complete set of data is not available for that flight, the *FTM Coprocessor* sends a request for flight data to the *FTM*. The *FTM* in turn sends the request to the *Traffic Model Functions* at the central site. When the flight data is returned to the *FTM*, it updates the flight table and passes the data to the *FTM Coprocessor.* The *FTM Coprocessor* will not return flight data to the requesting process until the missing flight data has been retrieved.

The *ASP* maintains the status and demand counts of all alerted elements for the next two and one quarter hours. The *ASP* obtains the alert data from alert summary files shipped from the central site. (The central site *ASP* sends the alert files to the field sites through the *ETMS Communications Functions.*)

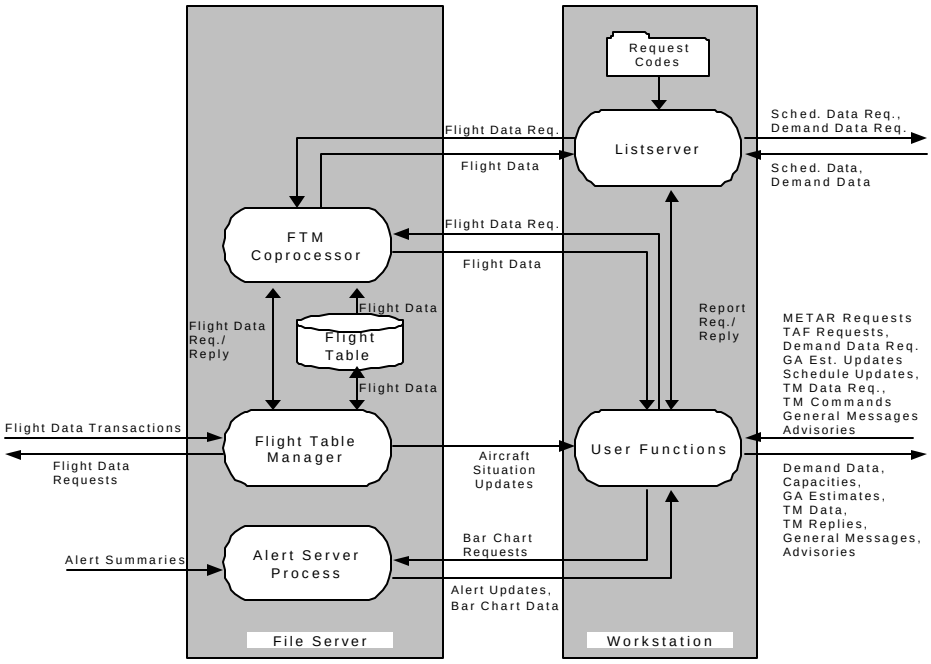

Figure 3-8. Data Flow of the User Support Functions – Field Site

The *Listserver* provides flight list, flight counts, arrival delay, time verification, fix loading, and list flight plan reports to the traffic management specialist. A traffic management specialist enters a request for a report through *User Functions*. The *User Functions* pass the request to the *Listserver*, which gets the required data, processes the data, and formats the requested report. The *Listserver* sends the report back to the *User Functions*, which present the report to the traffic management specialist.

The *Listserver* gets data for a report as follows. By default, if the requested report is for flight data within the next 12 hours, the *Listserver* first gets a list of the flights from the *Traffic Model Functions* at the central site. The *Listserver* then sends the flight list to the *FTM Coprocessor*, which fills in the detailed data for each flight in the list. If the requested report is for flight data beyond 12 hours, or if the user has specifically requested schedule data, the

Listserver gets the data from the *SDB Functions* at the central site. (The *SDB Functions* provide all the data for the report.)

Other *User Support Functions* (not shown in Figure 3-8) are *Weather Server* and *Delay Advisor*.

3.7 User Functions

There are several independent ETMS *User Functions*: the *TSD, Traffic Management Shell (TM Shell), Delay Manager, Electronic Mail (EMail), ETMS Log, Autosend, Router,* and *Route Manager*. The specialist can invoke many of these display functions by means of the *Tool Manager* (see Figure 3-9).

The *TSD* presents ETMS data to the traffic management specialist on a graphics display. The *TSD* allows the traffic management specialist to control the data displays in selective ways to enhance the usefulness of the data. The *TSD* provides displays of aircraft situation data, monitor/alert data, and weather data. The *TSD* gets data from the *FTM* and *ASP* at the field site and from the *Traffic Model Functions* and *SDB Functions* at the central site.

The *TSD* acquires data in two ways: automatically and through request/reply; the *Listserver* uses only request/reply.

When the *TSD* starts, it registers with the *FTM*. The *FTM* automatically sends aircraft situation updates every one minute to all registered *TSDs*. Similarly, when a traffic management specialist requests monitor/alert data, the *TSD* registers with the *ASP*. The *ASP* automatically sends alert updates to all registered *TSDs* whenever it receives an update from the central site (every five minutes). In both cases, the *automatic* data is sent to the *TSD* over the LAN through the *ETMS Communications Functions*.

The *TSD* and *Listserver* obtain all other data on request. The *TSD* requests flight data (for *alerted flights*) from the local *FTM Coprocessor*, SAs and FTs from the *Weather Server* (at the central site), and demand data (for monitor/alert reports and bar charts), capacities, and GA estimates from the *Traffic Model Functions* at the central site. (Bar chart information can come from either the local or the central site.) Similarly, the *Listserver* requests flight data from the local *FTM Coprocessor*, demand data from the *Traffic Model Functions* at the central site, and schedule data from the *SDB Functions* at the central site. All request/reply data transfers are performed through the *ETMS Communications Functions*.

The traffic management specialist can update data through the *TSD*. The *TSD* sends capacity, GA estimate, and alert status updates to the *Traffic Model Functions* at the central site, sends schedule database updates to the *SDB Functions* at the central site, and sends traffic management commands to purge ground delay programs to the *TM Functions*. For each update, the *TSD* receives a reply from the selected function either acknowledging the update or indicating an error in the update. The *TSD* sends the updates and receives the replies through the *ETMS Communications Functions*.

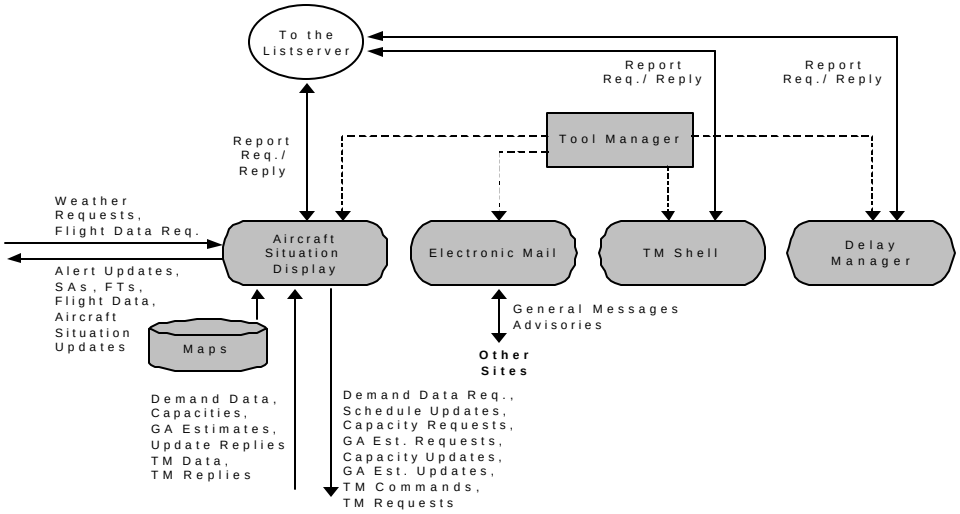

Figure 3-9. Data Flow of the ETMS User Functions – Field Site

The Traffic Management Shell (*TM Shell)* allows the specialist to execute many of the non-graphical commands available in the *TSD*, without any of the graphical displays (overlays, boundaries, flights, etc.) ever appearing. This gives the specialist access to the same data available in the full *TSD*, while allowing it to be used in a more efficient manner. Using the *TM Shell*, the specialist can issue individual commands and execute script files. *TM Shell* accesses the ETMS data in the same manner as the *TSD*.

Delay Manager simulates and evaluates the effects of controlling arriving flights for any selected airport. This function allows the specialist to evaluate the effects of implementing a ground stop program, to monitor the status of an EDCT program, and to simulate the releasing of flights from a ground stop or EDCT program. The *Delay Manager* accesses flight data through requests to the *Listserver*.

The *Electronic Mail (EMail)* allows the specialist to send advisories or general messages to addresses on the ARINC and NADIN networks, as well as to any other ETMS node. This function provides the ability to send one-line general messages, as well as detailed messages

composed of text and information from files that have been saved (e.g., advisories, flight reports, or message templates).

Other *User Functions* (not shown in Figure 3-9) are *Autosend* (see Section 3.5), *Router* (see Section 3.4), *Route Manager*, and *ETMS Log*.

3.8 Schedule Database Functions

The *Schedule Database (SDB) Functions* maintain an airline schedule database at the ETMS central site, feed scheduled flights through the *External Communications Functions* to the *Traffic Model Functions* for inclusion in the near-term databases, respond to requests for schedule data, and update the schedule database as instructed by traffic management specialists. The *SDB Functions* are composed of three processes, as shown in Figure 3-10. The *SDB Functions* exist only at the central site.

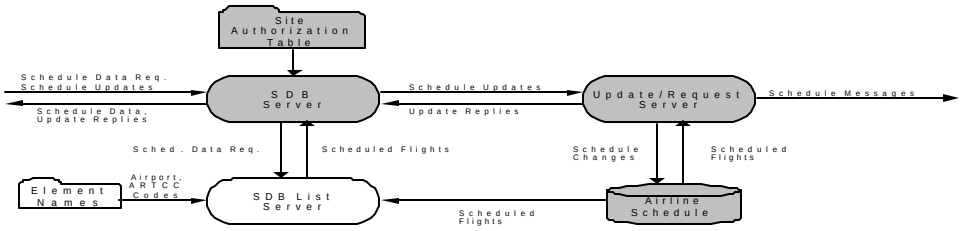

Figure 3-10. Data Flow of the Schedule Database Functions

The *SDB Server* handles all of the request/reply communications with other ETMS functions (i.e., all communications except the sending of schedule messages to the central site *User Support Functions*). The *SDB Server* receives schedule data requests and schedule updates through the *ETMS Communications Functions*. The *SDB Server* sends schedule updates to the *Update/Request Server*, first checking whether the update is from an authorized site. The *SDB Server* sends schedule data requests to the *SDB List Server*. When the *SDB Server* receives the schedule data back from the *SDB List Server*, it sends the data back to the requesting process through the *ETMS Communications Functions*. When the *SDB Server*

receives the update reply back from the *Update/Request Server*, the *SDB Server* returns the reply to the requesting process.

The *SDB List Server* receives schedule data requests from the *SDB Server*, looks up the requested flights in the airline schedule database, and returns the scheduled flights to the *SDB Server*. The schedule data request specifies airport(s) or ARTCC(s), whether to return arrivals and/or departures for the specified airport(s) or ARTCC(s), the date and time range for the flights, and a filtering ARTCC option. If the request is for an ARTCC, the *SDB List Server* returns all flights arriving at and/or departing all airports located within the specified ARTCC. If a filtering ARTCC is given, the *SDB List Server* checks each flight to see if the other end of the flight meets the filtering criteria. For example, if an arrival list is being prepared, the *SDB List Server* returns only those flights whose departure airport is within the filtering ARTCC.

The *Update/Request Server* feeds scheduled messages to the central site *External Communications Functions* and performs updates to the airline schedule database. The *Update/Request Server* automatically sends scheduled flights to the central site *External Communications Functions* when the flights' departure times fall within twelve hours of the current time. The *Update/Request Server* receives schedule updates from the *SDB Server* and makes the necessary changes to the airline schedule database. Schedule updates may cause a flight to be added, cancelled, inhibited, or activated. The *Update/Request Server* sends back a reply indicating either that the update was performed successfully or that an error was encountered. When a schedule update affects a flight scheduled to depart in the next twelve hours, the *Update/Request Server* also sends a schedule message to the *External Communications Functions*, indicating a flight has been added or cancelled.

The schedule messages are patterned to look like NAS messages. To add a scheduled flight, the *Update/Request Server* sends an *FS* message, which looks much like a NAS flight plan message (FZ). To cancel a scheduled flight, the *Update/Request Server* sends an *RS* message, which looks much like a NAS cancellation message (RZ). The *Update/Request Server* enhances the FS and RS messages by including the departure date of the flight. The departure date allows other ETMS functions to distinguish between occurrences of the same flight on different days. The schedule messages are passed to the *Traffic Model Functions* through the *External Communications Functions* so that they appear in the same stream of data as the NAS messages.

3.9 Traffic Model Functions

The *Traffic Model Functions* model the performance of flights; project the traffic demands at all airports, sectors, and fixes of interest to traffic management; and generate alerts for those airports, sectors, or fixes whose demand exceeds capacity. The *Traffic Model Functions* are shown in Figure 3-11.

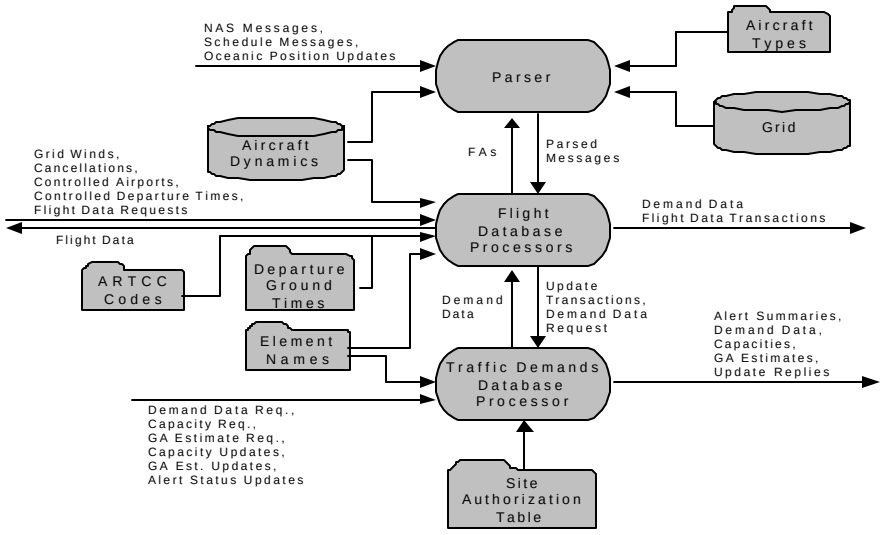

Figure 3-11. Data Flow of the Traffic Model Functions

The *Parser* receives the NAS messages, OMP messages, and schedule messages from the *External Communications Functions*, extracts the contents of the messages, and converts the contents to the internal format required for the subsequent processing. If a flight path (field 10) is contained in a message, the *Parser* interprets the flight path using the geographical data from the grid database and flight profile (i.e., ascent and descent) data from the aircraft dynamics database to produce a detailed, three-dimensional path for the flight. The internal description of the flight path, known as an *event list*, includes all events of interest to traffic management; i.e., airport arrivals and departures, sector entries and exits, fix crossings, airway (i.e., jet or Victor airway) entries and exits, and ARTCC entries and exits. The *Parser* also uses the aircraft types file to determine the *user class* (i.e., commercial, general aviation, military), *weight class* (i.e., small, large, heavy), and *aircraft type class* (e.g., single-engine piston prop) for the aircraft designator in the message. The *Parser* packages all information

extracted from the message and passes it to the *Flight Database (FDB) Processors*, which use the data to update the flight databases.

A NAS amendment message (AF) may not contain sufficient data for the *Parser* to produce an event list. For example, if an AF contains a field 10 but not an aircraft type, a new event list cannot be produced. In these cases, the *Parser* sends the data available in the AF to the *FDB Processors* without an event list. *The FDB Processors* then look up the needed data in the flight database and send it back to the *Parser*. The *Parser* then models the flight, produces an event list, and sends it back to the *FDB Processors*.

The *FDB Processors* maintain a database of all *active, proposed, scheduled,* and *controlled* flights for the next 12 hours and previous 12 hours.

> **NOTE:** Two identical *FDB Processors*, each processing approximately half of the flight data, are used to handle the computation load.

The *FDB Processors* build the flight database using the information received from the *Parser* and the controlled departure times and flight cancellations received from the *TM Functions*. The *FDB Processors* also maintain a database of current and predicted grid winds data extracted from the grid winds data received from the *Weather Server*. The *FDB Processors* use the flight data, the grid winds data, departure ground times, and data from the aircraft dynamics database to predict the event times for the flight. The *FDB Processors* re-compute a flight's event times whenever new data affecting the flight is received. Projected (future) event times are updated to actual (past) flight times as actual data is received in the NAS messages.

The *FDB Processors* send update transactions containing flight events and computed event times to the *Traffic Demands Database (TDB) Processor*, which uses the data to update the traffic demands database. The *FDB Processors* send an update transaction whenever a flight's events or event times change.

The *FDB Processors* also send flight data transactions containing data from the parsed messages and computed data through the *ETMS Communication Functions* to the central and field site *User Support Functions*, which use the data to update the local flight tables. The *FDB Processors* send flight data transactions automatically for each parsed message received or as a reply to data requested by an *FTM*.

The *FDB Processors* send flight data, either automatically or on request, to the *TM Functions*, which use the data to apply controlled departure times. The *FDB Processors* send flight data automatically when both a controlled departure time (CDT) and a flight plan (FZ) have been received for a flight (causing a CT to be sent by the *TM Functions*). The order in which the CDT and FZ are received does not matter. The *FDB Processors* also send flight data to the *TM Functions* in response to a flight data request.

The *FDB Processors* also support the *TM Functions* in the application of Fuel Advisory messages (FAs). When the *TM Functions* receive an FA, they send a transaction to the *FDB Processors* containing the controlled airport and the controlled time periods. Upon receipt of a controlled airport transaction, the *FDB Processors* send a demand data request to the *TDB Processor*, which returns an arrival list for the controlled airport and times to the *TM Functions* (which apply fuel advisory delays to any flights whose flight plans have already

been received). Flight plans received after this time are checked by the *FDB Processors* to see if a fuel advisory delay should be applied to the flight. If so, the *FDB Processors* send a flight data transaction to

the *TM Functions*, which determine the amount of delay that should be applied to the flight. The new controlled departure time for the flight is sent back to the *FDB Processors* just as an EDCT would be. Normal EDCT processing then continues in the *FDB Processors* and the *TM Functions.*

> **NOTE:** The processing of fuel advisory delays is embedded in the *FDB Processors* to ensure that the checking for existing flight plans and the checking for new flight plans is completely synchronized.

The *TDB Processor* uses the update transactions from the *FDB Processors* to maintain the predicted and actual traffic demands at each monitored airport, sector, and fix. The *TDB Processor* keeps counts of events for each element in 15-minute intervals or *buckets.* The *TDB Processor* also keeps the list of flights that comprise the demand in each bucket. The *TDB Processor* keeps separate counts by flight status (*scheduled, proposed, actual*) and can maintain demand data for 24 hours into the past and for up to 40 hours into the future.

When the *TDB Processor* updates a demand count, it checks the demand against the capacity. If the capacity is exceeded by the active flight count, the *TDB Processor* sets a flag, indicating a *red* alert for that time interval. If the capacity is exceeded by the proposed flight count, the *TDB Processor* sets a flag, indicating a *yellow* alert for that time interval. Every five minutes, the *TDB Processor* scans all the elements and generates alert summary files that contain the demand counts for all alerted elements. The *TDB Processor* sends the alert summary files to the local *ASP*, which distributes them to the field sites.

The *TDB Processor* can receive alert status updates entered by traffic management specialists at field sites. The *TDB Processor* first checks whether the update is authorized. If the update is from the ATCSCC, it is always authorized. Otherwise, the *TDB Processor* looks up the element specified in the update in the element names file to find the ARTCC associated with the updated element. The update is only allowed if it was generated at the element's ARTCC or a TRACON within that ARTCC. If an update is authorized, the *TDB Processor* sets the alert flag for the specified time interval to *previously red* or *previously yellow*. Once an alert status has been updated, the *TDB Processor* will not change it back to its previous color regardless of changes in the demand count or capacity value within that time interval. However, if the changes in the demand counts or capacity cause a previously yellow alert to go red, the alert status will be changed to red. After an alert status update is processed, the *TDB Processor* generates a reply, which indicates that the update was allowed or that an error was encountered.

The *TDB Processor* maintains *default* and *today's* capacity and GA estimate values for each element by 15-minute intervals. The default values are pre-defined and cannot be changed by the traffic management specialist. The *TDB Processor* initially sets today's values to the default values each day; it then changes them according to any capacity updates and GA estimate updates received from traffic management specialists. The *TDB Processor* checks the authorization of capacity and GA estimates updates in the same manner as for alert status updates. The *TDB Processor* uses today's capacities for determining alerts. When the *TDB Processor* performs a capacity or GA estimate update, it generates a reply, indicating either that the update was performed successfully or that an error was encountered.

The *TDB Processor* processes requests for demand data, capacities, and GA estimates. The demand data includes the demand counts, capacities, GA estimates, list of flight IDs, and flight event times for the requested element and time intervals. Capacity and GA estimate replies include both the default and today's values for the requested element and time intervals. Demand data requests for ARTCCs, airports, sectors, and fixes are processed by the *TDB Processor*.

If a demand data request is for one or more ARTCCs, the *TDB Processor* looks up all airports located within the ARTCC(s) and produces the demand data for all those airports. An ARTCC demand data request may be accompanied by one or more filtering ARTCCs/airports. If so, the *TDB Processor* assembles the demand data for the requested ARTCCs and passes the demand data with the filtering ARTCC(s)/airport(s) to the *FDB Processors*. The *FDB Processors* look up each flight and keep it only if it meets the filtering criteria (for example, a flight in an arrival list is kept if its departure airport is in a filtering ARTCC). The *FDB Processors* send the resultant, filtered set of demand data back to the requesting site.

3.10 Traffic Management Functions

The *Traffic Management (TM) Functions* process the EDCTs and FAs (from the ATCSCC) and SIs from the airlines along with flight data and demand data from the *Traffic Model Functions* to generate CTs to the NAS host computers. The *TM Functions* also maintain a history of all flight control data. The *TM Functions* consist of the *Controls Request Server,* the *Apply Ground Controls Process*, and the *Archive Controls Data Process.* They are shown in Figure 3-12.

The *Apply Ground Controls Process* receives EDCTs and FA delays from the ATCSCC and SIs from the airlines all via the *External Communication Functions*. The *Apply Ground Controls Process* checks SIs against rules determined by FAA policy, generates replies, and applies the legitimate SIs to previously received EDCTs. The flight control data (FAs and controlled departure times) are sent to the *Traffic Model Functions*. The *Traffic Model Functions* send flight data back to the *Apply Ground Controls Process* when a combination of a controlled time or delay and a flight plan exists. The *Apply Ground Controls Process* then generates a CT based on the flight data from the *Traffic Model Functions*. The *Apply Ground Controls Process* sends CTs and SI replies to the External Communication Function for transmittal to the NAS, the ATCSCC, and the airlines.

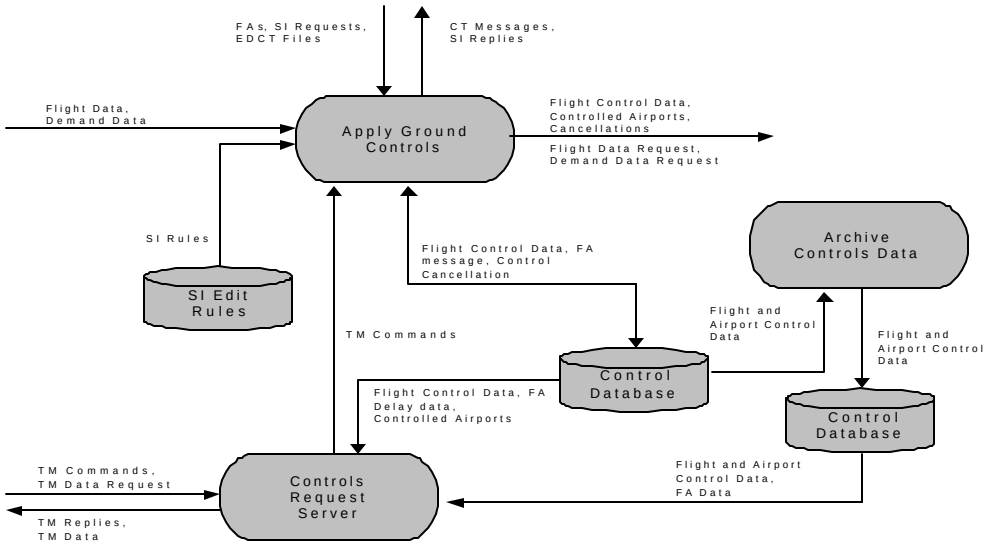

Figure 3-12. Data Flow of the Traffic Management Functions

The *Apply Ground Controls Process* also processes the TM Commands: purge, sub off, and sub on received from the ATCSCC (via the *ETMS Communications Function*). When a purge command is received, the *Apply Ground Controls Process* cancels previously received controls. The *Apply Ground Controls Process* also sends a reply to the ATCSCC acknowledging the purge command. When a sub off command is received, the *Apply Ground Control Process* rejects all subsequent SIs for the specified airport(s) until a sub on command is received. This process also receives TM commands from the *Controls Request Server*.

The *Controls Request Server* accepts TM commands and data requests from the *ETMS Communications Functions*, retrieves flight control data and FA data from the Control Database, and sends out TM data to the *ETMS Communications Functions*.

The *Archive Controls Data Process* receives flight and airport control data from the Control Database and saves a history in the Control Archive.

3.11 Auxiliary Support Functions

The *Auxiliary Support Functions* are needed to support the ETMS but do not run as part of the real-time processing. The *Auxiliary Support Functions* are used to convert external data into static files and databases used by the run-time ETMS. The *Auxiliary Support Functions* are run periodically. There are several *Auxiliary Support Functions*, as shown in Figure 3-13. Other *Auxiliary Support Functions* (not shown) are the *Ground Time Prediction Subsystem* and *Run-Time Support*.

The *Process Geographical Data* function processes the geographical data received from the NOS, the NFDC, and data files from the ARTCCs. The geographical data includes airport, NAVAID, and airspace fix locations; sector and SUA boundaries; runway layouts; and structured airway definitions.

The *Process Geographical Data* function reads the data from tapes received from NOS, NFDC, or the ARTCCs, extracts the necessary data, checks the data for errors, and processes the data into four specialized forms used by the run-time ETMS:

(1) Grid database — used by the *Parser* to interpret flight paths

(2) Maps database — used by the *TSD Display* to draw graphic display of geographical data on the screen

(3) Element names file — used by the *FDB Processors* and *TDB Processor* to convert element indexes to names before sending to *User Support Functions*

(4) Airports file — used by the *FTM* to *ghost* aircraft towards their destination airports

The *Process Geographical Data* function also receives OAG airport definitions from the *Build Airline SDB* function. The *Process Geographical Data* function ensures that all airports referenced in the airline schedule database are defined in the maps and grid databases.

The *Process Geographical Data* function provides data to two other *Auxiliary Support Functions*. The FAA airport definitions are sent to the *Build Airline SDB* function. The pacing airport, monitored airport, arrival fix, and ARTCC names are sent to the *Build Request Codes* function.

The *Build Airline SDB* function processes the airline schedule data file received from the OAG and produces the airline schedule database used to support the *SDB Functions*. The *Build Airline SDB* function reads the schedule data from the OAG data file, converts it to internal formats, checks for errors, assigns flight paths to the scheduled flights, and builds a set of indexes to the scheduled flights to support the types of access (e.g., by departure time, by airport) required during run time. The *Build Airline SDB* function receives the flight paths from the *Determine Flight Paths* function and assigns them to the scheduled flights based on the city pair and airline. The *Build Airline SDB* function also provides a list of the OAG airline designators to the *Build Request Codes* function and a list of OAG aircraft designators to the *Process Aircraft Type Data* function.

The *Determine Flight Paths* function uses past flight plans (FZs) to derive flight paths, cruising speeds, and cruising altitudes to be assigned to scheduled flights. The *Determine Flight Paths* function analyzes a recent sample (at least one week's worth) of flight plans to determine the most commonly filed flight plan data for each airline between each city pair. The resultant data is sent to the *Build Airline SDB* function for assignment to the scheduled flights. The method for determining flight plan data for scheduled flights is further described in Appendix A of the *ETMS Functional Description.*

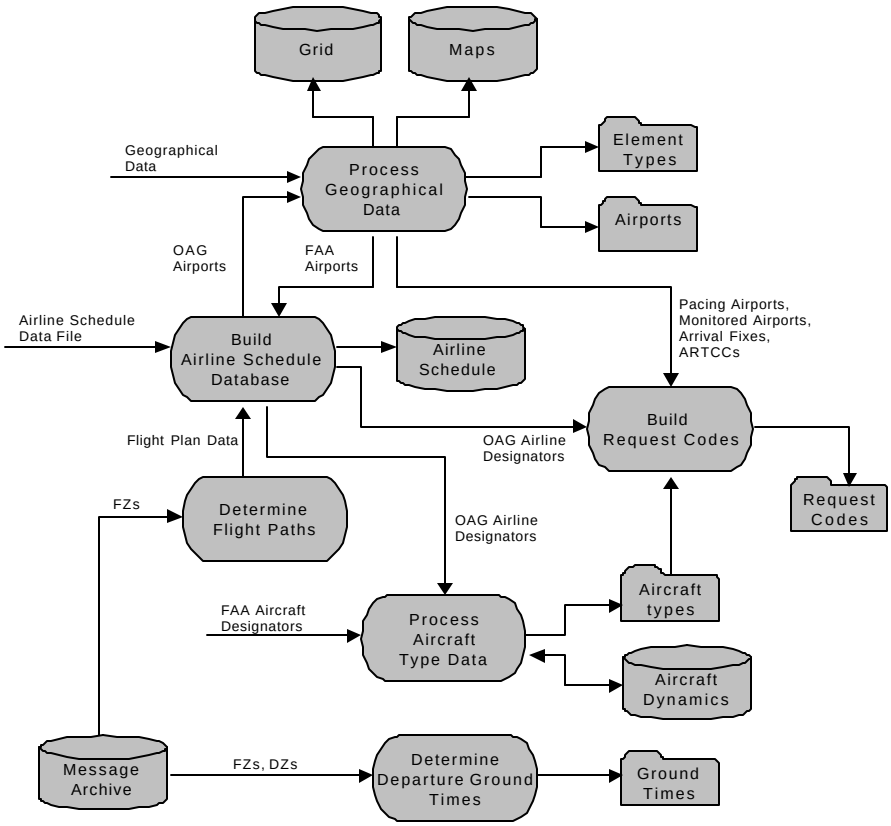

Figure 3-13. Data Flow of the Auxiliary Support Functions

The *Process Aircraft Type Data* function updates the aircraft types file with new aircraft designators. The aircraft types file contains the user class, weight class, and aircraft type class for each aircraft designator. The aircraft types file is used by the *Parser* to determine the classes for individual flights; it is also used by the *Build Request Codes* function.

The *Build Request Codes* function uses data from several of the other *Auxiliary Support Functions* to produce a file of *codes* that can appear as part of a report request made to the *List Server*. The *Build Request Codes* function combines pre-defined, special-use words (e.g., **list, count, info**) with the following data:

- Pacing airport names, monitored airport names, arrival fix names, and ARTCC names produced by the *Process Geographical Data* function
- Airline designators extracted from the OAG schedule data by the *Build Airline SDB* function
- Aircraft designators from the aircraft types file produced by the *Process Aircraft Type Data* function

The *Determine Departure Ground Times* function derives ground time data from archived historical flight data and is described in Appendix B of the *ETMS Functional Description.*

The *Process Aircraft Type Data* function updates the aircraft dynamics database when new aircraft designators are defined by the FAA, or when aircraft designators that are not defined by the FAA appear in the OAG schedule data. The aircraft dynamics database contains a set of pre-defined ascent and descent profiles. When a new aircraft designator is defined, the *Process Aircraft Type Data* function assigns an ascent and descent profile to the aircraft type, based on parameters such as weight class, engine type, maximum ascent rate, and maximum descent rate. The aircraft dynamics database is used by the *Parser* and *FDB Processors* for flight profile and flight time modeling.

3.12 Data Distribution and Recovery

The ETMS system design directly impacts the response time for data displays after a request is made. The response time for a data request is most heavily affected by the physical location of the data.

For the purpose of understanding the response time to a data request, it is sufficient to consider the data to be in three places: the traffic management specialist's workstation, the *file server* at the traffic management specialist's site, and the ETMS central site. Data on the traffic management specialist's workstation can be accessed most quickly, since no communication is required (i.e., data is on a disk or in physical memory). Data on the file server at the site cannot be accessed as quickly for two reasons: the data must be communicated over the *local area network* (LAN) and the file server may be busy performing other functions. Data access from the central site is generally slowest, because the data must

be communicated over landlines or satellite links, which operate at much slower speeds than a LAN.

Figure 3-14 illustrates the physical locations of the ETMS databases. The only ETMS data stored directly on the traffic management specialist's workstation is the map overlay data. The flight data used to generate the aircraft situation updates is stored on the file server at each site, as is the alert summary data and some of the weather data (all except METAR and TAF reports). All other data accessible through the *TSD* is stored on nodes at the central site.

3.12.1 Flight Transaction Messages

Flight data is transmitted from the central site to the field site file server by means of *flight transaction messages*. There are various types of flight transaction messages. These include position messages, route messages, time messages, block/altitude messages, TZ messages, cancel messages, total traffic management (TTM) messages, and critical messages. The last two types, TTM and critical, are used for data recovery.

3.12.2 Field Site Data Recovery

Situations can arise where the flight data at a field site is incomplete. The file server can detect such a situation in two ways. First, when the file server processes are first started up, they may discover that the flight tables are empty or that the flight tables may not have been updated for some time period. Second, when the file server provides flight data for a flight list generated at the central site, it may discover that it is missing data for a flight in the list. In either case, the file server will recover missing data from the central site databases. The file server can request recovery data in several ways:

- A complete recovery is requested when the file server has no flight data.
- A partial recovery is requested when the file server is missing data for a time range.
- A flight data recovery is requested when the file server is missing data for a specific flight or set of flights.

In each case, the requested data is transmitted from the central site. The central site sends the data through the normal flight data transactions that are used to send current flight data.

3.12.3 Central Site Data Recovery

Situations can arise at the central site where one of the redundant processing strings becomes non-functioning. In such cases, a support services specialist can initiate a recovery process, which uses the functioning string as the data source to repopulate the non-functioning string.

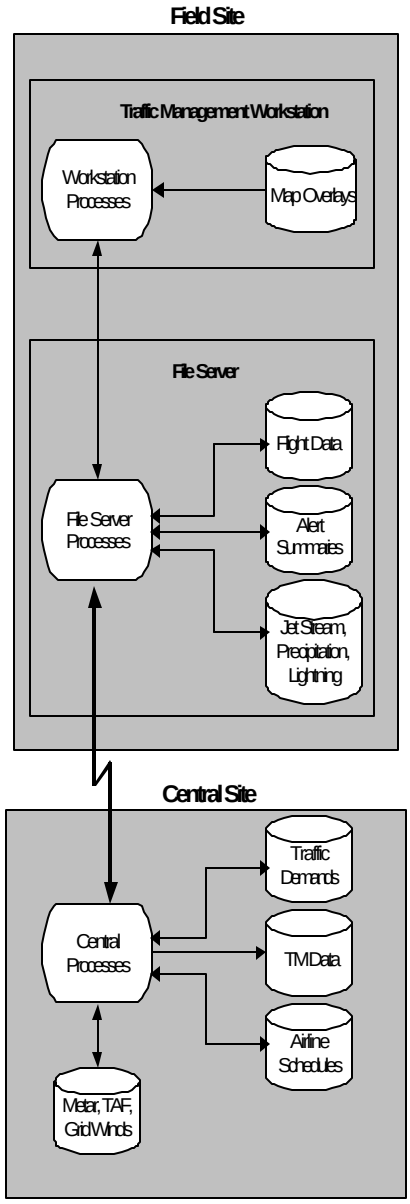

Figure 3-14. ETMS Database Distribution